IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Serial No. 10/611,372
Filing Date 6/30/03
Confirmation No. 3151
Inventorship ..... Soluk et al.
Assignee ..... Microsoft Corporation
Group Art Unit 2153
Examiner ..... Yasin M Barqadle
Attorney's Docket No. MS1-1575US
Title: ..... Method and Apparatus for Configuring a Sever

DECLARATION UNDER 37 C.F.R. §1.131

The undersigned inventor hereby declares as follows:

1. My residence, post office address and citizenship are as stated below next to my name.

2. I am an inventor of the invention described and claimed in U.S. Patent Application Serial No. 10/611,372, entitled "Method and Apparatus for Configuring a Server".

3. I conceived of the claims disclosed in the above-referenced patent application prior to January 7, 2002, the filing date of U.S. Patent Publication Number 2003/0131078 A1 to Scheer (hereinafter "Scheer"). Attached hereto as **Exhibit A** are the claims associated with my invention, which I conceived prior to January 7, 2002.

4. My invention is documented by correspondence and invention disclosure documents prepared prior to June 30, 2003. Attached hereto as **Exhibit B** is a redacted copy of a disclosure document. This document provides a

description of the invention and drawings. Non-essential portions of Exhibit B have been redacted. Although some of the actual date(s) have been redacted from Exhibit B, I declare that the claimed subject matter, as recited in Exhibit A, was conceived prior to January 7, 2002, and reduced to practice in April 2002, as evidenced by Exhibit B. For clarity, Exhibit B supports claims 1-34 of the instant application, as presented in Exhibit A.

5. Based in part on my own knowledge and also on information and belief, between the conception of the invention disclosed in the above-referenced application and the date of reduction to practice of the invention in April 2002, the inventors and/or other support staff worked diligently to reduce the invention to practice. More specifically, during this period, the activities directed to reducing my invention to practice were performed on an approximately continuous basis between the conception of the invention and the reduction of the invention to practice in April 2002, as at least further evidenced below:

   I. The redacted portions of a correspondence created in November 2001 and submitted as **Exhibit C**. The correspondence is an email between one or more inventors and other parties discussing the progress of the invention.

   II. The redacted portions of a correspondence created in December 2001 and submitted as **Exhibit D**. The correspondence is an email between one or more inventors and the other parties discussing the need to evaluate certain portions of the invention.

III. The redacted portions of a correspondence dated January 2002 and submitted as **Exhibit E**. The correspondence is an email between one or more inventors and other parties discussing modifications of the invention.

IV. The redacted portions of a correspondence dated February 2002 and submitted as **Exhibit F**. The correspondence is an email between one or more inventors and other parties discussing modifications of the invention.

V. The redacted portions of a correspondence dated March 2002 and submitted as **Exhibit G**. The correspondence is an email between one or more inventors and other parties discussing modifications to a prototype of the invention.

VI. The redacted portions of a correspondence dated April 2002 and submitted as **Exhibit H**. The correspondence is an email between one or more inventors and other parties discussing usability testing of the prototype of the invention.

I certify that all statements made herein of my own knowledge are true, and that all statements made on information and belief are believed to be true; and further, that these statements were made with the knowledge that the making of willfully false statements and the like is punishable by fine or imprisonment, or both, under section 1001 of Title 18 of the United States Code, and may jeopardize the validity of any patent issuing from this patent application.

* * * * * * * * *

Full name of inventor: Kirk Soluk

Inventor's Signature _____ Date: 1-30-08

Residence: Ann Arbor, MI

Citizenship: USA

Post Office Address: c/o Microsoft Corporation, One Microsoft Way, Redmond, WA 98052

* * * * * * * * *

Full name of inventor: Everett McKay

Inventor's Signature _____ Date: _____

Residence: Redmond, WA

Citizenship: USA

Post Office Address: c/o Microsoft Corporation, One Microsoft Way, Redmond, WA 98052

* * * * * * * * *

Full name of inventor: Hitesh Raigandhi

Inventor's Signature _____ Date: _____

Residence: Redmond, WA

Citizenship: USA

Post Office Address: c/o Microsoft Corporation, One Microsoft Way, Redmond, WA 98052

* * * * * * * * *

Full name of inventor: Kirk Soluk

Inventor's Signature _____ Date: _____

Residence: Ann Arbor, MI

Citizenship: USA

Post Office Address: c/o Microsoft Corporation, One Microsoft Way, Redmond, WA 98052

* * * * * * * * * *

Full name of inventor: Everett McKay

Inventor's Signature Everett McKay Date: Jan 30, 2008

Residence: Redmond, WA

Citizenship: USA

Post Office Address: c/o Microsoft Corporation, One Microsoft Way, Redmond, WA 98052

* * * * * * * * * *

Full name of inventor: Hitesh Raigandhi

Inventor's Signature _____ Date: _____

Residence: Redmond, WA

Citizenship: USA

Post Office Address: c/o Microsoft Corporation, One Microsoft Way, Redmond, WA 98052

* * * * * * * * *

Full name of inventor: Kirk Soluk

Inventor's Signature _____ Date: _____

Residence: Ann Arbor, MI

Citizenship: USA

Post Office Address: c/o Microsoft Corporation, One Microsoft Way, Redmond, WA 98052

* * * * * * * * * *

Full name of inventor: Everett McKay

Inventor's Signature _____ Date: _____

Residence: Redmond, WA

Citizenship: USA

Post Office Address: c/o Microsoft Corporation, One Microsoft Way, Redmond, WA 98052

* * * * * * * * * *

Full name of inventor: Hitesh Raigandhi

Inventor's Signature Hitesh Raigandhi Date: 2/6/08

Residence: Redmond, WA

Citizenship: USA

Post Office Address: c/o Microsoft Corporation, One Microsoft Way, Redmond, WA 98052

* * * * * * * * * *

Full name of inventor:	Yang Gao
Inventor's Signature	_____ Date: 01/29/08
Residence:	Issaquah, WA
Citizenship:	P.R.China
Post Office Address:	c/o Microsoft Corporation, One Microsoft Way, Redmond, WA 98052

* * * * * * * * * *

Full name of inventor:	Praerit Garg
Inventor's Signature	_____ Date: _____
Residence:	Kirkland, WA
Citizenship:	USA
Post Office Address:	c/o Microsoft Corporation, One Microsoft Way, Redmond, WA 98052

* * * * * * * * * *

Full name of inventor:	Yang Gao
Inventor's Signature	_____ Date: _____
Residence:	Issaquah, WA
Citizenship:	P.R. China
Post Office Address:	c/o Microsoft Corporation, One Microsoft Way, Redmond, WA 98052

* * * * * * * * * *

Full name of inventor:	Praerit Garg
Inventor's Signature	Praerit Garg Date: 2/4/08
Residence:	Kirkland, WA
Citizenship:	USA
Post Office Address:	c/o Microsoft Corporation, One Microsoft Way, Redmond, WA 98052

Exhibit A

1. (Original) A method comprising:

identifying at least one role associated with a target server;

identifying one or more services associated with the role;

identifying one or more ports associated with the role;

presenting the identified services and ports associated with the role to a user; and

requesting the user to select among the identified ports for activation in the target server.

2. (Original) A method as recited in claim 1 wherein the identified services and ports are limited to those that are relevant based on information obtained from a knowledge base.

3. (Original) A method as recited in claim 1 wherein the identified services and ports are limited to those that are relevant based on information regarding a target server.

4. (Original) A method as recited in claim 1 further comprising activating the selected services and ports.

5. (Previously Presented) A method as recited in claim 4 wherein at least one of the services associated with the role and the ports associated with the roles are identified from a knowledge base.

6. (Previously Presented) A method as recited in claim 1 further comprising:

identifying an operating system level of a target server;

determining one or more security levels for the target server based on the identified operating system level of the target server; and

selecting one of the determined security levels for the target server,

wherein identifying at least one role includes identifying at least one role associated with the target server based on the selected security level.

7. (Previously Presented) A method as recited in claim 1 further comprising deactivating unselected services and ports.

8. (Original) A method as recited in claim 1 further comprising generating an output file containing services and ports selected by the user.

9. (Original) A method as recited in claim 1 further comprising displaying details regarding the role in response to a request by the user.

10. (Original) A method as recited in claim 1 further comprising displaying a list of options for handling a service associated with the target server that is not defined in a knowledge base.

11. (Original) A method as recited in claim 10 further comprising requesting the user to select an option for handling the service.

12. (Original) One or more computer-readable memories containing a computer program that is executable by a processor to perform the method recited in claim 1.

13. (Previously Presented) A method comprising:

identifying one or more roles associated with a target server;

identifying one or more services associated with the roles;

displaying the identified services associated with the roles;

allowing a user to modify the displayed services; and

identifying the modified services as active services and identifying unmodified services as inactive services.

14. (Original) A method as recited in claim 13 wherein identifying services associated with the role includes retrieving data from a knowledge base.

15. (Original) A method as recited in claim 13 further comprising generating an output file containing services modified by the user.

16. (Original) A method as recited in claim 13 wherein the user is responsible for configuring the target server.

17. (Original) A method as recited in claim 13 further comprising generating an output file identifying active ports and inactive ports.

18. (Original) One or more computer-readable memories containing a computer program that is executable by a processor to perform the method recited in claim 13.

19. (Previously Presented) A method comprising:

identifying a role associated with a target server;

identifying one or more ports associated with the role;

presenting the identified ports associated with the role to a user;

requesting the user to select among the identified ports associated with the role; and

identifying the selected ports as active ports and identifying ~~the~~ unselected ports as inactive ports.

20. (Original) A method as recited in claim 19 further comprising generating an output file identifying ports selected by the user.

21. (Original) A method as recited in claim 19 wherein the one or more ports are identified using information contained in a knowledge base.

22. (Original) A method as recited in claim 19 wherein the user is responsible for configuring the target server.

23. (Original) A method as recited in claim 22 further comprising:

displaying one or more ports associated with the role; and

requesting the user to select among the one or more ports to activate in the target server.

24. (Original) One or more computer-readable memories containing a computer program that is executable by a processor to perform the method recited in claim 19.

25. (Original) An apparatus comprising:

a pre-processor to receive information regarding server roles from a knowledge base and to receive characteristics of a target server, wherein the pre-processor generates a file containing server role information relevant to the target server, and wherein information in the file regarding services and ports associated with the server roles is presented to a user for selection; and

a configuration engine coupled to the pre-processor, wherein the configuration engine configures the target server based on the user's selection of services and ports.

26. (Original) An apparatus as recited in claim 25 further comprising a user interface application to generate an output file identifying services selected by the user.

27. (Original) An apparatus as recited in claim 25 further comprising a user interface application to generate an output file identifying ports selected by the user.

28. (Original) An apparatus as recited in claim 26 wherein the configuration engine applies the output file when configuring the target server.

29. (Original) An apparatus as recited in claim 27 wherein the configuration engine applies the output file when configuring the target server.

30. (Original) One or more computer-readable media having stored thereon a computer program that, when executed by one or more processors, causes the one or more processors to:

identify a role associated with a target server;

identify one or more services associated with the role;

identify one or more ports associated with the role;

display the identified services and ports associated with the role; and

receive selected services and ports to be activated on the target server.

31. (Original) One or more computer-readable media as recited in claim 30 wherein the one or more processors further activate the selected services and ports during configuration of the target server.

32. (Original) One or more computer-readable media as recited in claim 30 wherein the one or more processors further deactivate unselected services and ports during configuration of the target server.

33. (Original) One or more computer-readable media as recited in claim 30 wherein the one or more processors further identify the one or more services and the one or more ports associated with the role are identified from a knowledge base.

34. (Original) One or more computer-readable media as recited in claim 30 wherein the one or more processors further display one or more options for handling a service associated with the target server that is not defined in a knowledge base.

Exhibit B

Microsoft Patent Pre-disclosure Document

Note: Hover the pointer over underlined section headings for more help and click on a link for a sample document.

This section contains items that are required to get the patent process under way

Document Author (name and email): Kirk Soluk (kirksol)

Title of Invention: **Secure Server Roles** **MS File # (if known): 167538.1**

Inventor(s): Kirk Soluk, Praerit Garg, Vishnu Patankar, Jin Huang, Everett McKay, Hitesh Raigandhi, Yang Gao, Shawn Wu

Introduction: SSR is an extensible software tool designed to help customers improve the security of their Windows 2000 and Windows 2003 servers. [redacted]

Strategic Importance:

[redacted]

Microsoft Patent Pre-disclosure Document

Note: Hover the pointer over underlined section headings for more help and click on a link for a sample document.

[redacted]

Description of the invention, Diagrams & Flowcharts:
Here are exerpts from [redacted]

1 Architecture

1.1 Architectural Overview

The Secure Server Roles initiative consists of four primary components:

- **Knowledge Base** – The knowledge base component identifies operating system and application parameters that need to be configured or analyzed from a security perspective. It then abstracts these pieces of data and information into functional terms that can be understood by technical professionals even though these professionals may not be Windows security experts. The knowledge base also contains UI "directives" that help determine the default values rendered by the UI under various circumstances. Separate knowledge bases exist for both Windows 2000 Server (Win2kKB.xml) and .Net Server (DotNetKB.xml).

- **UI –** User input must be provided in order to maximize security for a given deployment scenario. The SSR UI, implemented in the form of a Wizard, leverages the knowledge base (previously described) in conjunction with the current state of the system in order to solicit functional requirements from the user and derive a resultant security policy. Assuming the knowledge base is accurate, the security policy (if applied) will improve system security without sacrificing the specified functional requirements. The UI persists the security policy as a text-based (XML) file which can then be interpreted and applied by an infrastructure component.

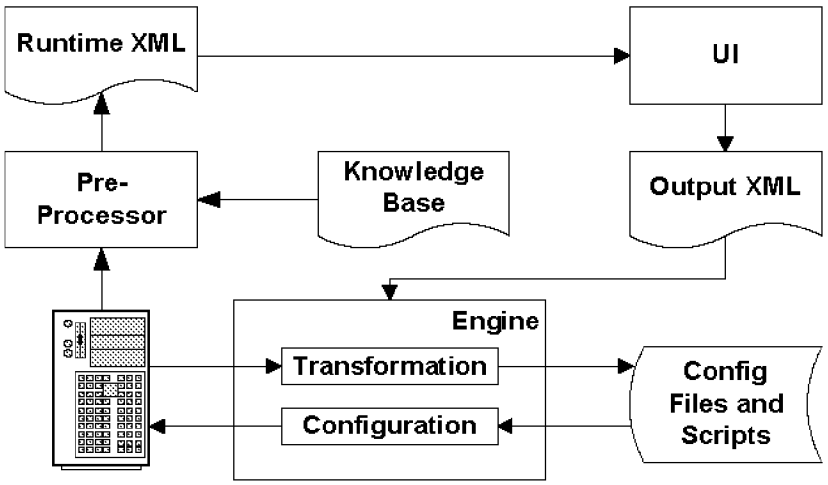
- **Pre-processor** – The pre-processor creates a *runtime-specific knowledge base* by reconciling the *raw* knowledge base (mentioned above) against the current state of the target system. It is the run-time knowledge base that is actually consumed by the UI component. This pre-processing stage allows the UI to render more apropriate information and to start with more accurate defaults. The fact that the pre-processing stage is run outside of the UI-proper is an implementation detail that significantly improves the architecture from a coupling standpoint (makes the coupling looser) and thus from a test and development perspective as well.

- **Back-end Engine** - The back-end engine (or simply *engine)* provides a framework and a set of components that can transform the *output XML* (produced by the UI) into native scripts and configuration files then subsequently apply those scripts and settings to the system during a configuration phase. The back-end engine also supports rollback and analysis operations in addition to configuration.

The primary relationships between these high-level architectural components is illustrated here:

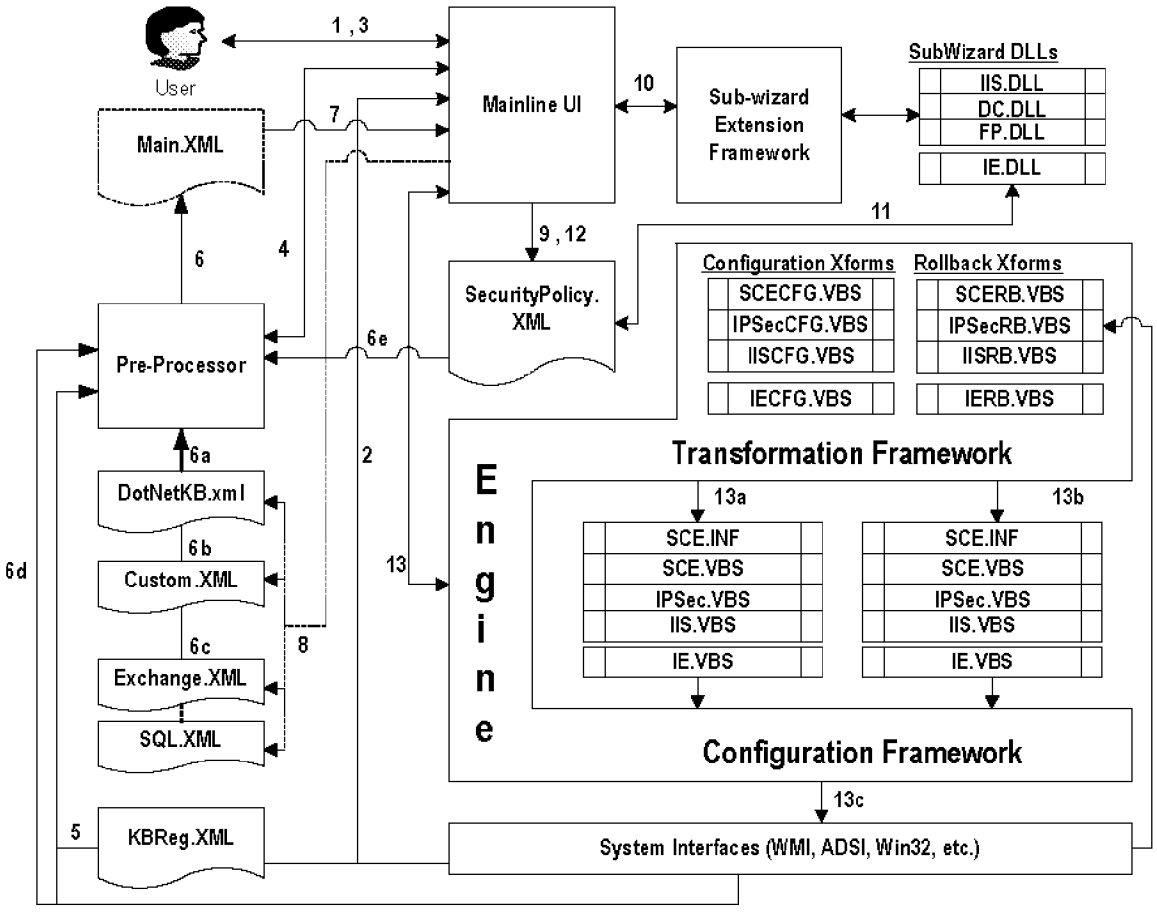
Microsoft Patent Pre-disclosure Document

Note: Hover the pointer over underlined section headings for more help and click on a link for a sample document.



1.2 Architectural Details

The following diagram provides a more detailed view of the SSR architecture and data flow. It describes the processes, inputs, and outputs used to create and apply a security policy. Integration points for extensions are also described.:



[redacted]

Microsoft Patent Pre-disclosure Document

Note: Hover the pointer over underlined section headings for more help and click on a link for a sample document.

ssrroot
- \CommonLib
- \ConfigureFiles
- \KBs
- \Logs
- \Members (we should change this to Registration if we need it at all – see Win2k section)
- \Policies
- \ReportFiles (we should change this to Reports)
- \RollbackFiles
- \TransformFiles (we should change this to Transforms)

where ssrroot is chosen by the user at installation time and contains the ssr executables. This root can be determined by examining the following registry value which is also populated by setup:

KeyPath: HKLM\Software\ Microsoft\Windows NT\CurrentVersion\Secwiz\
Value: Path
Type: REG_SZ
Data: Absolute path to SSR root directory (i.e. No environment variables).

If this registry value does not exist, the UI should critically fail.

Assuming SSR has been installed and registered correctly, the UI prompts the user for a target system then offers the following options:
- Create a new security policy
- Modify an existing security policy
- Apply an existing security policy
- Rollback the previously applied security policy
- Analyze the system for compliance with an existing policy

The following steps describe what happens when the user chooses to create, then apply a new security policy.

1. The UI receives three primary pieces of input from the user:
   a) The operation the user wants to perform (e.g. Create a new security policy)
   b) The path to the knowledge base directory that should be used (optional)
      - By default, KBReg.xml is located in the ssrroot\KBs directory of the *local* machine. The user can specify an alternate knowledge base directory by submitting a UNC, DNS, or IP Address-based path.
   c) The "target" server that should be used to build the security policy

2. The UI queries the OS level of the target system then opens KBReg.xml to determine the set of security levels that are supported by the knowledge base associated with the target OS.

3. The UI presents the available security levels to the user. The security level chosen by the end-user drives the subsequent default settings rendered by the wizard. By default, SSR specifies two security levels: Maximum and Typical. An admin can customize SSR to change the displaynames, descriptions, order of appearance, and default selection for these two values. The admin can also customize SSR to automatically pick either Maximum or Typical in which case the end-user is not presented with a choice.

4. Once the UI has determined the target machine and the supported security levels, it calls the pre-processor passing it:
   a) the target computer name
   b) the path (local or remote UNC) to the knowledge base directory that should be used
   c) the chosen security level

Note that the target computername and the path to the knowledge base directory have nothing to do with each other. The UI receives progress feedback from the pre-processor as the proceeds through the following steps.

5. The pre-processor opens up KBReg.XML from the local or remote directory location which it received from the UI. This must be the same KBReg.xml that the UI previously opened when the UI determined the security levels. The pre-processor uses KBReg.XML in conjunction with the OS version of the target machine in order to determine the *raw*, *custom*, and *extension* KBs that should be used to build main.xml.

6. The pre-processor combines information from several sources in order to create a machine-specific KB (**main.xml**) that will drive the "mainline" (Service and Port configuration) portion of the UI. Main.xml is generated each time the user creates a new policy or modifies an existing one. Inputs to the pre-processor include:

    a) **RAW KB (DotNetKB.xml or Win2kKB.xml)** –The "raw" KBs, specify OS-specific role, service and port relationships as well as information related to other OS-Specific (e.g. Registry Value) settings. RAW KB's should not be modified by the customer otherwise changes will be lost on update. Instead customers should customize and extend the raw kb using custom and extension KBs as described below.
    b) **Custom KB (Custom.XML)** – The custom KB provides a mechanism by which customers can override definitions that already exist in the raw KB. For example, a customer may wish to include an extra service or choose different defaults when maximum or typical is chosen. In order to accomplish this, the customer re-defines the abstraction (e.g. a role) in an alternate file and registers that file as a customization in kbreg.xml. If a custom KB contains a definition that is not in the raw KB, it is acceptable for the pre-processor to add that definition.
    c) **Extension KB's (Exchange.XML, SQL.XML,…)** – Extension KBs provide a mechanism by which customers can add definitions to the raw KB. For example, SSR will ship Extension KB's for Exchange and SQL Server. This will cause the UI to render these additional roles and to enable the appropriate services and ports just like any other role. KB Extensions may or may not have sub-wizards associated with them. For example, in the diagram above, **SQL.XML** specifies service and port information for SQL server, but there is no SQL-specific subwizard that executes as a result of selecting the SQL server role. Extension KB's allow services and ports to be configured for any 3rd party application in a completely declarative manner.
    d) **Local or Remote System** – The pre-processor queries the target system (passed to it by the UI) in order to identify numerous settings that, in turn, make the UI "smart". For example, the pre-processor queries the target system to determine what services are installed, what their current startup mode is, what NICS are present, what IP addresses\subnets are associated with each NIC, whether or not the machine is joined to a domain, what ports are active etc. This information is passed on, in one form or another, to the UI via main.xml.
    e) **SecuritPolicy.XML** – TBD. If the user has chosen to modify a policy that was previously created, then the pre-processor will likely be involved to generate another main.xml. In this case, the pre-processor will likely need to override its normal default settings with those specified by the previously created template.

    The order of precedence for resolving conflicts between the system and various knowledge bases is, (from highest precedence to lowest precedence) as follows:

    1. System
    2. Custom KB(s)
    3. Raw KB
    4. Extension KBs

If there are conflicts within a given level. For example, two extension KB's both define the SQL Server role, then the extension registered last in kbreg.xml wins.

7. Machine-specific information, satisfiable roles, extensions, customizations, default recommendations, pointers to sub-wizards etc. are all available for the mainline UI through **main.xml**. Main.xml is always generated on the local machine in ssrroot\kbs. The Mainline UI uses this information to help the user interactively maximize security for a given deployment scenario.

8. Cut from v1. The UI allows the user to update raw, custom or extension KBs (including creating new kb's if necessary) to include information discovered by the pre-processor or further refined by the user. For example, if the pre-processor finds that Lotus Notes is running it will add the service to main.xml (since it's not defined in the raw KB). Subsequenltly the user may declare that Notes uses a specific port rather than any random port as assumed by the pre-processor. The UI could create an extension KB for Lotus Notes so that the next time the wizard is run, the user does not have to declare the specific port again. Without this feature, the user will have to either manually update the appropriate KB which is a reasonable expectation for v1.

9. Once the mainline UI has completed, but before any application-specific subwizards are invoked, it stores the OS-specific security policy in some temporary file. The stored policy can then be used by any application-specific subwizards in order to determine any previous settings that may have been selected. For example, if the server is not a print server, the IIS-specific subwizard can infer that Internet Printing cannot be supported.

10. After the OS-specific settings have been stored, the UI invokes any *applicable* application-specific subwizards. There are two types of application-specific subwizards:
    a) **Role-based subwizards** are launched as a result of a role being directly or *indirectly* selected. If a role has been selected (either directly or indirectly) and main.xml indicates that the role has an application-specific subwizard, and the subwizard is installed and registered with the UI, then the UI invokes the application-specific subwizard. In the architecture diagram above, IIS.DLL, DC.DLL, and FP.DLL are all role-based sub-wizards corresponding to the Web Server, Domain Controller and File\Print Server roles defined in the raw KB (DotnetKB.xml).
    b) **Role-independent subwizards** are launched regardless of the roles selected by the user. Role-independent subwizards are responsible (perhaps by querying some system settings) for determining whether or not they themselves should be rendered. In the architecture diagram above, **IE.DLL** is an example of a role-independent subwizard. There is no IE role defined in any of the KB's.

    All role-based subwizards should be invoked before any role-independent subwizards. As of now, sequencing within a class of wizards is not a requirement. One must be able to create a sub-wizard using Visual Basic as a programming language.

See ntspecs\security\management\ssr\???? for the subwizard extension framework details including how subwizards are registered and associated with roles. (**Eric Brown)**

11. Sub-wizards will need to be able to query and update the security policy thus far created in order to optimize it's defaults. When in edit mode, sub-wizards will need to be able to query both the previous policy that is being edited AND the new policy that is being created as a result of the edit. Note that the mainline wizard does not need to query the "old" policy directly because the information from the "old" policy is already incorporated into main.xml via the pre-processing function.

Edit mode should be interpreted as using the previously created policy to determine the default selections for a new policy that is being created from scratch. While this is the default for the Mainline wizard (as a result of using the pre-processor), this concept needs to be extended to the sub-wizards to avoid scenarios such as the following: User edits a policy and deselects a role that has a sub-

Microsoft Patent Pre-disclosure Document

Note: Hover the pointer over underlined section headings for more help and click on a link for a sample document.

wizard. In this case, we do not want to persist the old sub-wizard data. Similarly, if the role is left in tact, the sub-wizard should be able to see the previous defaults.

[redacted]

12. After the last role-based or role-independent sub-wizard has completed, the mainline UI stores the cummulative policy using a filename provided by the user, then asks the use if they want to apply that policy to the target system...

13. The UI exposes a single "configuration" option to the user. This translates into three sub-operations for the engine. Note: The sub-operations described below are exposed through "DoActionVerb", which is a method on the SSREngine Interface. Details are available in **ntspecs\security\management\ssr\SSREngine_II.doc**

    a) The XML security policy must be transformed into native configuration files and scripts that can be applied to the target system. This phase is performed by the Transformation component of the engine (by calling DoActionVerb with Prepare + Configure). The Transformation engine (TE) drives a set of registered components (i.e. scripts, executables, or XSLT transforms) that are capable of translating some abstract portion of the security policy xml into detailed native configuration files or scripts that can be directly applied to the system where the transformation took place.
    b) Rollback configuration files and scripts must be generated. This is also performed by the Transformation component of the engine (by calling DoActionVerb with Prepare + Rollback). Note that the rollback transforms must query the system in order to determine what values will change. The transformed configuration and rollback scripts are allowed to have the same names because they exist in different directories. It is not a requirement, however, that they have the same filenames.
    c) The configuration files and scripts generated by the transformation phase must be applied to the system. This is performed by the configuration component of the engine framework (by calling DoActionVerb with Apply + Configure).

It is important to note that generating the rollback scripts is not a pre-requisite for configuring the machine. Automatically generating the rollback scripts when the user requests a configure operation is an implementation decision made by the UI. Also, the UI does not proceed with configuration (step 3) if rollback generation (step 2) fails. The command line tool should allow more granularity.

Date of Conception: [redacted] **Date Reduced to Practice: April 2002**

[redacted]

[redacted]

Additional required info that is helpful and will save you time to provide now

[redacted]

Microsoft Patent Pre-disclosure Document

Note: Hover the pointer over underlined section headings for more help and click on a link for a sample document.

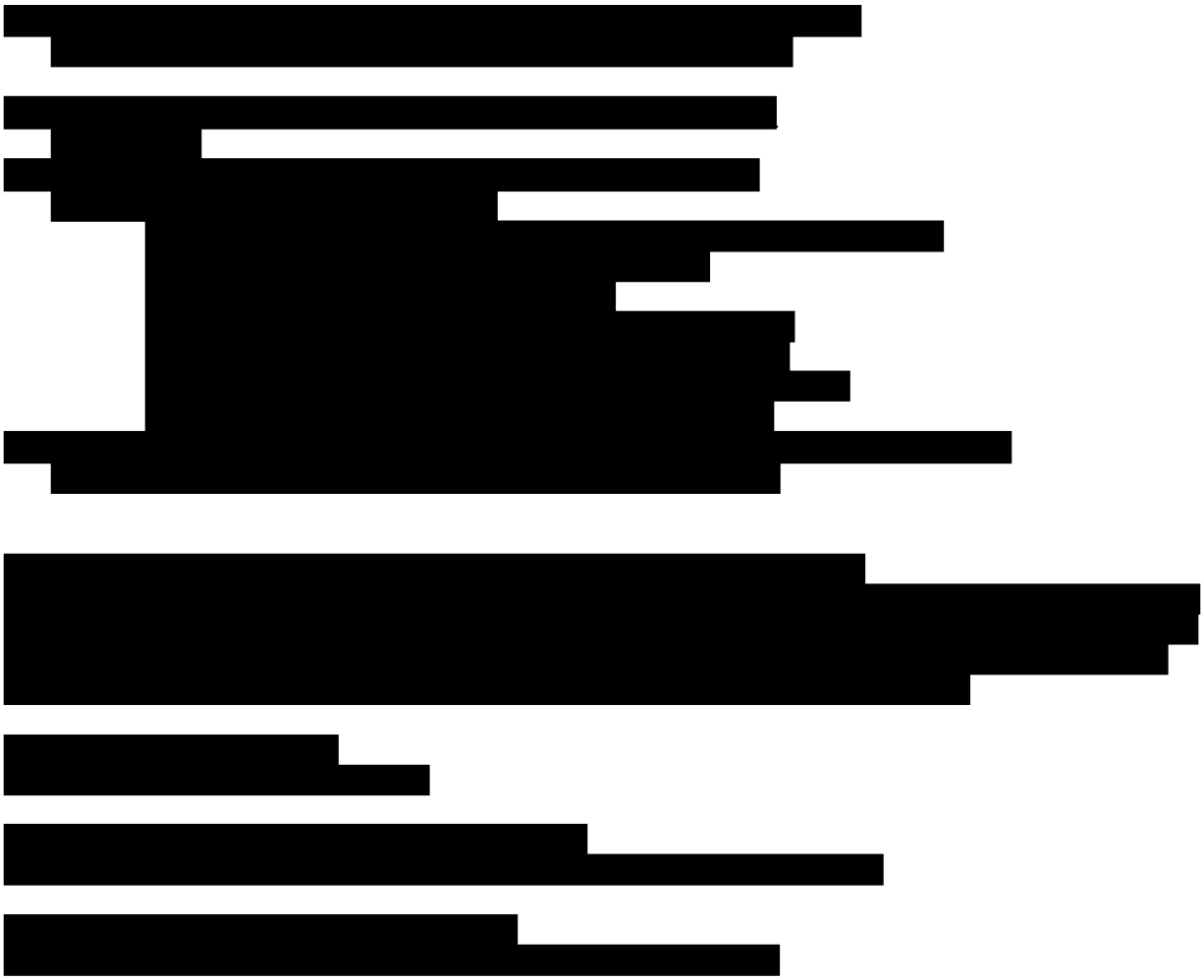


Exhibit C

From: Dana Huang [redacted]
Sent: Wednesday, November [redacted], 2001 12:19 AM
To: Vishnu Patankar; Everett McKay; Eric Kool-Brown; Yang Gao; Deepika Agarwal (Volt); Shawn Wu; Nick Finco
Cc: Lara Sosnosky; Kirk Soluk; Jason Garms
Subject: Update of SSR meeting today.

We followed the agenda below for today's meeting. Below is the summary of what we discussed:

Milestone 1 (11/15) status

UI Spec is mostly done with minor changes to finish. Everett was not in the meeting to confirm how much is the work left. The changes are not blocking dev or test at this time.

UI Devs are working on RC1 bugs so SSR wizard will probably be delayed for 2 weeks (11/30). Most of the pages are done without linking to the backend logic yet. The current UI code is in admin depot, under ssrui directory.

Engine spec and development for milestone was already done by 10/31. Security policy schema needs to be updated to include roles information. [redacted]

Test

UI test spec is in the progress. Deepika will have a rough estimate on testing effort for services/roles.
[redacted]

Customization of SSR UI

[redacted]

W2K

[redacted]

Other features

Exhibit C

[redacted]

Jin

-----Original Message-----
From: Jin Huang
Sent: Tuesday, November [redacted], 2001 12:03 AM
To: Eric Brown; Yang Gao; Everett McKay; Deepika Agarwal (Volt); Shawn Wu; Vishnu Patankar; Nick Finco
Cc: Lara Sosnosky
Subject: Let's have the SSR meeting today (Tuesday)

[redacted] I have a few things to discuss with everyone, including:

Milestone 1 status for both UI and engine
Test spec progress
Discuss possiblility of customized wizard pages driven by KB (will provide more detail)
Discuss W2K retarget
Then if we still have time, I want to discuss a few features.

Please mark your time and show up in the meeting.

Thanks,
Jin

Exhibit D

From: Kirk Soluk [redacted]
Sent: Saturday, December [redacted], 2001 10:35 PM
To: Everett McKay; Vishnu Patankar; Secure Server Roles Team
Subject: RE: reminder: we will have the weekly SSR meeting tomorrow at 4pm /6083

In the meantime, further info below...

-----Original Message-----
From: Everett McKay
Sent: Thursday, December [redacted], 2001 2:53 PM
To: Kirk Soluk; Vishnu Patankar; Secure Server Roles Team
Subject: RE: reminder: we will have the weekly SSR meeting tomorrow at 4pm /6083

We need to perform an evaluation of the resulting roles and tasks. [redacted]

-----Original Message-----
From: Kirk Soluk
Sent: Tuesday, December [redacted], 2001 2:39 PM
To: Vishnu Patankar; Secure Server Roles Team
Subject: RE: reminder: we will have the weekly SSR meeting tomorrow at 4pm /6083

I just checked in code complete OS service "schema" and a lot more "live" data: [redacted]

These files have the changes discussed in the attached mails along with definitions for: [redacted]

Summarizing the changes: [redacted]

Exhibit D

[redacted]

I don't know of any more schema changes that need to be made, but let me know if you see any problems. I'll continue to add more data and update SSR.Doc as well. [redacted]

-----Original Message-----
From: Vishnu Patankar
Sent: Monday, December [redacted], 2001 9:40 AM
To: Secure Server Roles Team
Subject: RE: reminder: we will have the weekly SSR meeting tomorrow at 4pm /6083

Nick/Deepika

Please bring questions regarding test to the meeting (I remember you had questions regarding this) since we need some momentum going on this. We can add test milestones to the schedule to get follow through and the big picture.

Vishnu

-----Original Message-----
From: Jin Huang
Sent: Monday, December [redacted], 2001 8:07 AM
To: Eric Brown; Yang Gao; Everett McKay; Vishnu Patankar; Shawn Wu; Nick Finco; Lara Sosnosky; Deepika Agarwal (Volt)

Exhibit D

Cc: Jason Garms
Subject: reminder: we will have the weekly SSR meeting tomorrow at 4pm /6083

We haven't had SSR meetings for a while due to conflict schedules. Let's have this week's meeting as scheduled to discuss project status and any issue. Please bring status with you if you have tasks past due or due in December. Particularlly, I am looking at the following:

-----Original Message-----
From: Jin Huang
Sent: Monday, December [redacted], 2001 8:54 AM
To: Eric Brown; Yang Gao; Kirk Soluk; Vishnu Patankar; Shawn Wu
Subject: Reminder: SSR deliverables for December (this month)

According to the SSR schedule, the following tasks should be done in this month. Please keep the schedule in mind – we need to be on schedule in order to get the project done in time. If you think there are problems reaching the goal, please let me know (specifically for the tasks that are past due or close to due).

Thanks - Jin

Task (working days)	Due Date	Duration Owner(s)

Exhibit D

Exhibit E

From: Shawn Wu [mailto:shawnwu@windows.microsoft.com]
Sent: Thursday, January [redacted], 2002 4:15 PM
To: Secure Server Roles Team
Subject: spec updated SSREngine_II.doc update.

I just updated the SSR backend engine's spec to reflect Kirk's decision. This change includes:

(1) Layout changes (names of the folders) and the creation of two rollback folders.
(2) Name collision avoidance rules.
(3) Feedback message definition and its meanings and variant's type and values.
(4) Multiple rollback folders and its use.

[redacted]

If you find any inconsistencies or inaccurate information, please let me know.

Thanks,

Shawn

Exhibit F

From: Kirk Soluk [mailto:IMCEAEX-_O=MICROSOFT_OU=FIRST+20ADMINISTRATIVE+20GROUP_CN=RECIPIENTS_CN=KIRKSOL@bf.umich.edu]
Sent: Friday, February [redacted], 2002 9:08 AM
To: Eric Brown; Secure Server Roles Team
Subject: RE: SSR UI Private Checkin

The spec (page 18 in ssr.doc) and the checked-in sample main.xml both say <selected> for roles.:

<Selected> *indicates whether the role is selected by default when the role UI first appears.*

Of course the user can change the default status of the <selected> field but you keep track of such changes in memory. Finally, when all desired roles (and now tasks) have been "selected" then all the contained services should implicitly be enabled - which means they have a startup type of manual or automatic.

<select> was previously used to indicate which services the UI should "select" as a result of the user selecting some role. But since we introduced tasks we do not need to do this anymore – we just pick all services associated with the selected roles and tasks. So there is no longer a <select> item associated with individual services that are contained within a role or task.

> -----Original Message-----
> **From:** Eric Brown
> **Sent:** Friday, February [redacted], 2002 8:37 AM
> **To:** Kirk Soluk; Secure Server Roles Team
> **Subject:** RE: SSR UI Private Checkin
>
> Sorry, omitted a couple of words below. The sentence should read:
>
> > Auto-enable services **from Independent Roles and any other Roles** whose Selected value is TRUE. (the spec says it should be Select, not Selected).
>
> This is by far the most important thing for me to do, so that will be next on my plate.
>
> *EricB*
> Commercial User Experience Development, Server Product Group
>
> > -----Original Message-----
> > **From:** Kirk Soluk
> > **Sent:** Friday, February [redacted], 2002 8:24 AM
> > **To:** Eric Brown; Secure Server Roles Team
> > **Subject:** RE: SSR UI Private Checkin
> >
> > Thanks Eric, some comments\clarifications below...
> >
> > > -----Original Message-----
> > > **From:** Eric Brown

Exhibit F

Sent: Thursday, January [redacted], 2002 8:19 PM
To: Secure Server Roles Team
Subject: RE: SSR UI Private Checkin

I just checked in my changes to fix most of the blocking problems. Vishnu, I also copied SecWiz.* to your MSI share.

Observations:

[redacted]

Things left for me to finish for this milestone:

[redacted]

Things to do for a later milestone:

[redacted]

EricB

Commercial User Experience Development, Server Product Group

Exhibit F

-----Original Message-----
From: Eric Brown
Sent: Thursday, January [redacted], 2002 6:37 PM
To: Yang Gao
Cc: Secure Server Roles Team
Subject: RE: SSR UI Private Checkin

Thanks, I've synced it and it builds fine. Now I need to fix my bugs as outlined in Kirk's email and the meeting today.

EricB

Commercial User Experience Development, Server Product Group

-----Original Message-----
From: Yang Gao
Sent: Thursday, January [redacted], 2002 6:28 PM
To: Eric Brown
Cc: Secure Server Roles Team
Subject: SSR UI Private Checkin

Eric,

I just checked in my new modifications.

Yang

Exhibit G

From: Stephan Betz [redacted]
Sent: Monday, March [redacted], 2002 3:24 PM
To: Everett McKay
Subject: RE: SSR Input

Yes, that would be just great. If you could get that to me in the morning it would even be fantastic. [redacted]

Thanks a lot,

Stephan

-----Original Message-----
From: Everett McKay
Sent: Monday, March [redacted], 2002 10:03 AM
To: Kirk Soluk; Stephan Betz
Subject: RE: SSR Input

I could have this for you tomorrow. I'm already booked solid for today. Is that OK?

-----Original Message-----
From: Kirk Soluk
Sent: Monday, March [redacted], 2002 9:01 AM
To: Stephan Betz
Cc: Everett McKay
Subject: RE: SSR Input

[redacted]

Everett, any chance we can add [redacted] from the previous prototypes to the current prototype so that we can give people a better idea of the complete picture?

/Kirk

-----Original Message-----
From: Stephan Betz
Sent: Friday, March [redacted], 2002 9:30 PM
To: Kirk Soluk
Subject: SSR Input

[redacted]

Exhibit G

Exhibit H

From: Ryan West [redacted]
Sent: Wednesday, April [redacted], 2002 8:11 AM
To: Everett McKay
Subject: RE: SSR Usability

Hi Everett –
Of course! [redacted]

Thanks
-Ryan

-----Original Message-----
From: Everett McKay
Sent: Tuesday, April [redacted], 2002 10:02 PM
To: Ryan West
Subject: SSR Usability

We should start planning on doing some type of usability testing for SSR. I suppose sometime in June would be ideal. Is this possible?